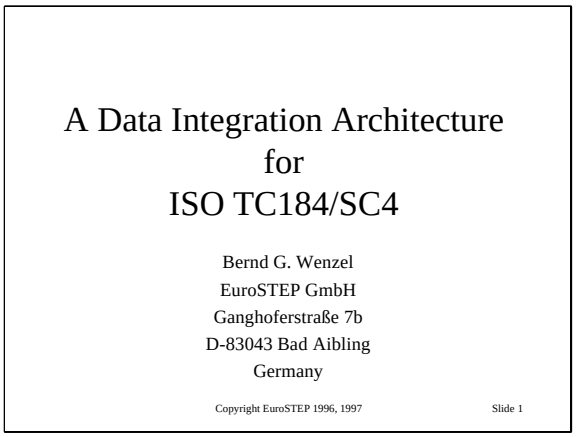
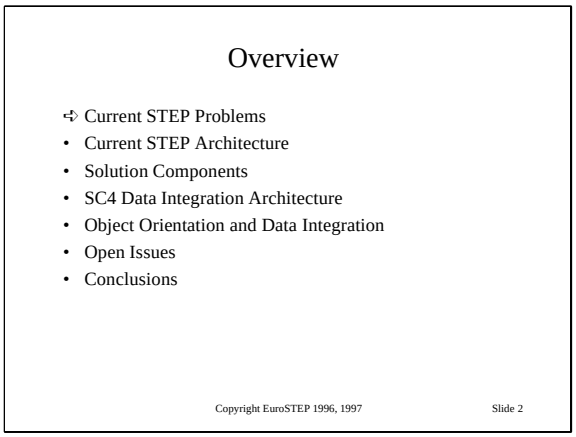


This set of slides proposes an evolution of the STEP data architecture, suited to support all standards developed under SC4, and beyond.

Is is based on:

- the ANSI SPARC architecture,
- the theory of Object Orientation,
- contributions to the STEP architecture made before, and
- personal experience.



In particular, this presentation will cover:

Current STEP Problems - the most urgent problems with STEP caused by the current STEP Architecture/Methodology

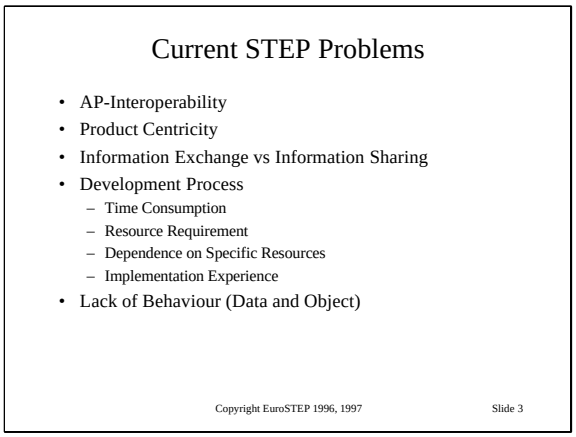
Solution Components - some basic technologies, on which the next edition of the STEP Architecture/Methodology will be based

SC4 Data Integration Architecture - the proposed data integration architecture

Object Orientation and Data Integration - some considerations about the evolution of the proposed architecture into the direction of Object Orientation

Issues - issues not covered by the data integration architecture and its object-oriented extension as outlined above.

So let’s start with the current problems of STEP.



The current STEP Architecture/Methodology is based on some assumptions regarding the industrial requirements, that turned out to be wrong later-on. The following problems have been identified as being caused, at least partially, by those wrong assumptions:

AP-Interoperability

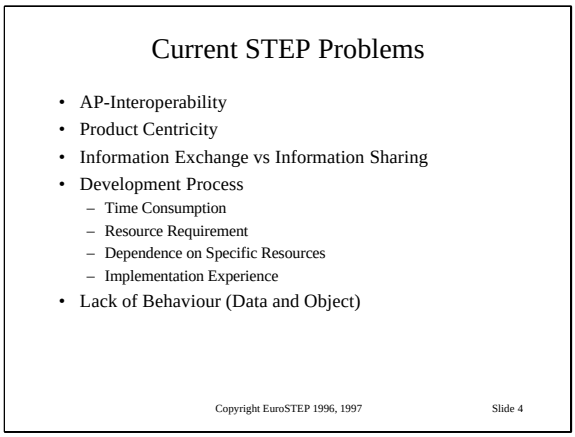
The current STEP Architecture/Methodology does not support the exchange of information beyond the limits of a single AP, at least not to the degree required by industry. Several industry initiatives, especially around PDES Inc. and ProSTEP are working on limited, short-term solutions for this issue. WG10 has been chartered to develop a generic long-term solution.

Product Centricity

The current STEP Architecture/Methodology requires, that all STEP data must be formally connected to a product. This creates problems, when we want to use the STEP Architecture/Methodology outside of the product data domain.

Information Exchange vs Information Sharing

The current STEP Architecture/Methodology supports the exchange of information - based on exchange files - rather well. At the same time, it prevents the sharing of STEP data by multiple applications in a logically integrated database, because it requires the information models to be overconstrained for this purpose.



Development Process

Time Consumption - the current STEP Architecture/Methodology requires, that 2 information models are developed, one dependent on the other, together with a mapping specification between them. This is considered a waste of time by many AP development projects.

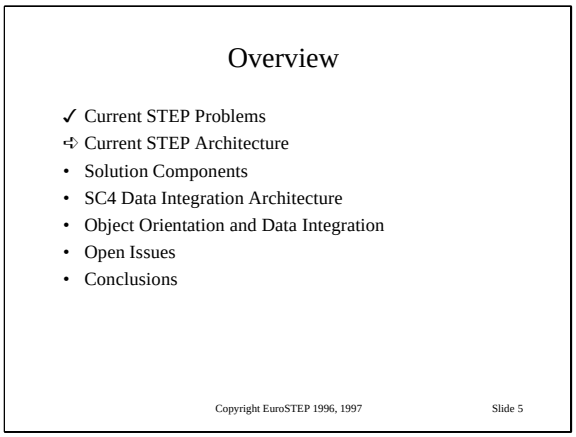
Resource Requirement - the current STEP Architecture/Methodology requires the definition of the information requirements in both formal and textual form. This is considered unnecessarily resource intensive by some AP development projects, especially regarding the structural requirements.

Dependence on Specific Resources - only a very limited number of people is able and authorized to perform some of the functions required by the STEP Methodology. This creates an unnecessary dependence on those resources.

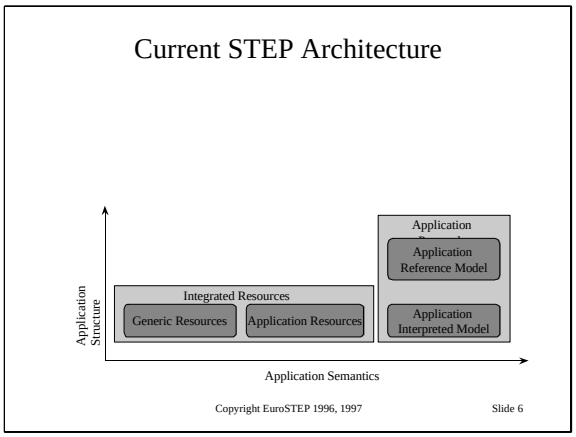
Implementation Experience - the current STEP Architecture/Methodology states, that all implementations shall be based on the AIM, the second information model to be created. This delays implementation feedback unnecessarily.

Lack of Behaviour

At present, STEP is only standardising data structures. The behaviour, both of the data objects and of the real world objects represented by the data objects, is completely ignored. Both are on the other side requirements for future data exchange and data sharing standards, such as STEP.



Now we will look at the current architecture for STEP. This will explain at least to some extent, where the currently percieved STEP problems come from.



In the current STEP Data Architecture, we find the following kinds of models:

Integrated Resource Models (IR) - Information models, which describe the information requirements for product data on a structurally highly abstract level. They consist of **Generic Resource Models (GIR),** which are thought to be totally independent of any specific application domain, and **Application Resource Models (AIR),** which are common for a family of application domains.

The genericity of GIRs is of course a consequence of STEP's limited context to product data. And even there it can be debated, whether for instance configuration control is required for all kinds of products.

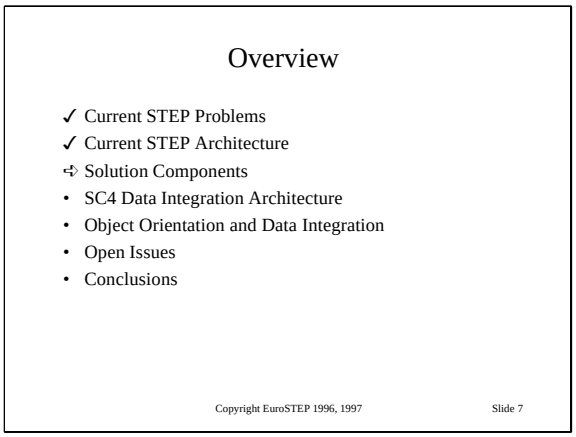
Application Reference Models (ARM) - Information models, which describe the information requirements for a particular domain (AP) in structures familiar to the application domain experts.

Application Interpreted Model (AIM) - Information models, which "interpret" the Integrated Resource Models, use the data structures defined in the Integrated Resources to fulfill the information requirements specified by an ARM.

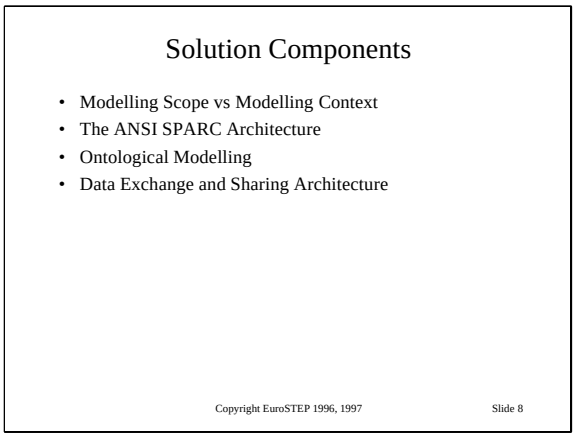
An **Application Protocol (AP)** is a standard consisting of an AAM, an ARM, and an AIM.

Application Activity Models (AAM) - to identify the scope of an AP.
These models are not shown on this slide.

The gap between the Integrated Resources and the Application Protocols symbolises the well-known problem of **AP-Interoperability**. We can see, that this architecture allows APs to share data structures irrespective of whether they underlying semantics are compatible or not.

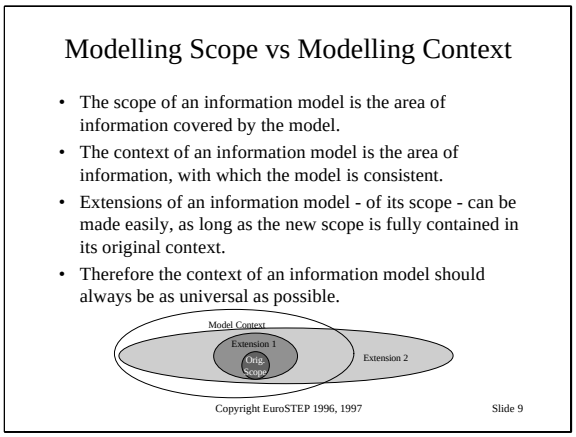


It is now the time to look at the components of our proposed solution.



It is worth noting at this point, that all these components are known since several years, some for decades, and are well tested in industrial applications.

The first part of this presentation concentrates on the data part of the new STEP architecture. The aspects of behavioural modelling will be handles separately, later on.



The distinguishing factor between an information model, where scope and context coincide, and one with the same scope but a much wider context is the information structures used. The wider the scope, the more primitive or normalised information structures will be found. Therefore we may say, that every information model with a universal scope consists of semantically irreducible information structures.

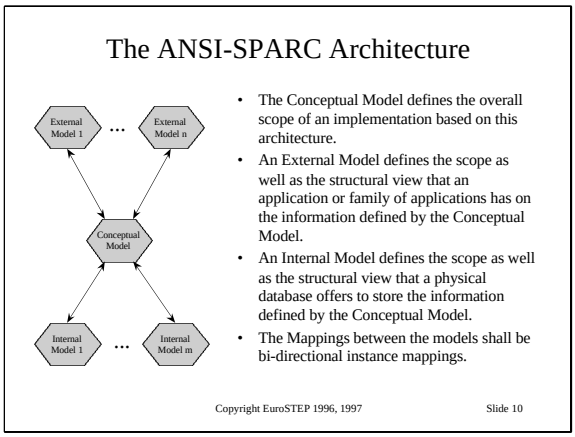
Explanation of the diagram:

The information model is originally defined with the **orig. scope** within its **context**.

Extension 1 will not create any problems, as its is totally included in the **context** of the information model.

Extension 2 on the other side may create lots of trouble, as it goes beyond the original **context** of the information model.

Note: Traditionally, the scope and the context of an information model are more or less identical, especially when the information model was intended to support implementation. As a consequence, the applications built based on those models are rather monolithic and hard to integrate.



In the context of this presentation, the term “model” is used to represent a set of constructs, which together completely provide the intended functionality. Nevertheless, this functionality may be too abstract to be implemented in isolation, without combining it with other models.

The term “schema” is used for the language representation of a syntactical subdivision of a model. Consequently, a model may consist of any number of schemas, but a schema always belongs to a single model.

ANSI-SPARC-Architecture

Other terms for this architecture are **3-Schema Architecture** and **3-Layer Architecture.**

Conceptual Model

Other terms for Conceptual Model are **Logical Schema/Model** or sometimes **Ontological Schema/Model**.

The term “Ontological Model” is more often used for a conceptual model with the most universal context possible. In the light of what has been said before, this is what we really need. Therefore it is best described using semantically irreducible information modelling constructs. This guarantees, that it is extensible over time with no (realistically minimal) impact on existing applications.

External Model

Another term for External Model is **Application Schema/Model**.

There is no requirement, that all External Models together fully cover the scope of the Conceptual Model. Therefore, the set of applications supported by a single Conceptual Model may grow over time.

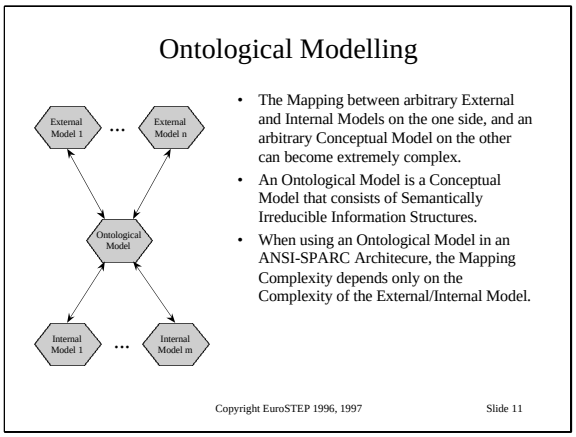
Internal Schema

Other terms for Internal Model are **Storage Schema/Model** or **Physical Schema/Model**.

There is no requirement, that all Internal Models together fully support the scope of the Conceptual Model. Therefore, the physical databases supporting a single Conceptual Model may grow over time, together with the application requirements.

External and Internal Model

There is no relationship between the number of External Models for a Conceptual Model and Internal Models for the same Conceptual Model. The different characters ”n” and ”m” have been used to indicate this.

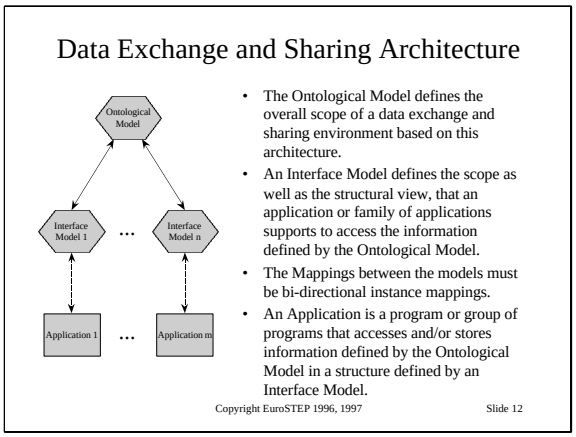


Oversimplified Analogy

In general terms, the complexity of the mapping between 2 arbitrary information models depends of the complexity of the 2 information models. It can be seen as the product between the complexity of the 2 information models.

The complexity of a truly semantically irreducible information model is by definition 1.

Therefore the complexity of the mapping of an arbitrary schema onto an Ontological Model is the complexity of the model to be mapped.



Ontological Model

As explained on the previous slide, the term ”Ontological Model” means a conceptual model consisting of semantically irreducible information structures only.

The main difference between these 2 architectures is, that the main direction of the flow of control (from External Models through the Ontological/Conceptual Model to the Internal Models) no longer exists.

Interface Model

An Interface Model is in principle an External Model and an Internal Model at the same time, or in modellling terms a common supertype of both.

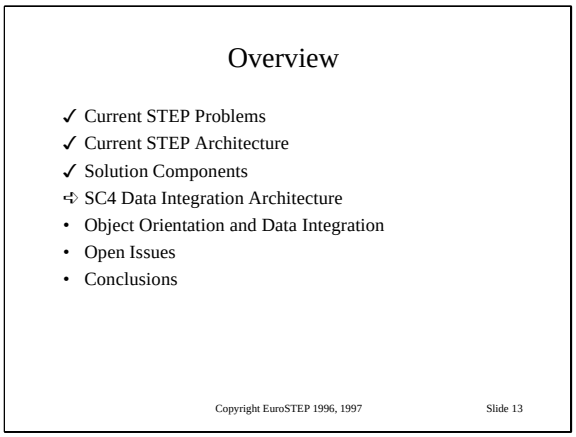
There is no requirement, that all Interface Models together fully cover the scope of the Ontological Model. Therefore, the set of applications supported by a single Ontological Model may grow over time.

Application

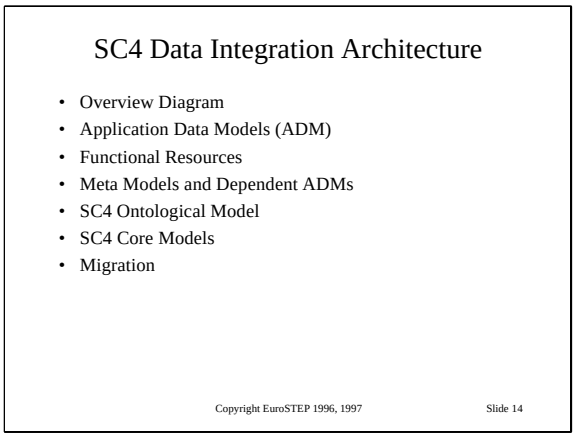
This term is used here in a very generic way. By intent, a datadase system is an application according to this definition.

Interface Model and Application

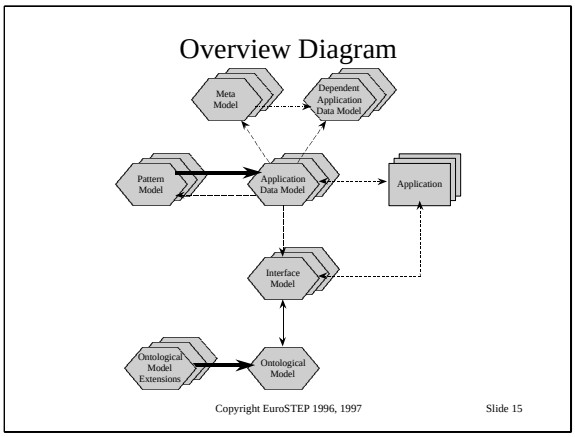
There is no relationship between the number of Interface Models for an Ontological Model and the number of Applications they support. The different characters ”n” and ”m” have been used to indicate this.



And now it’s time to start with the presentation of the new data integration architecture.



We will first see an overview of the new architecture, showing how all components fit together, before we will closer examine the different components.



Line Types:

Integration

Bi-directional Model Mapping

Derivation

Information Flow between Application and Interface Model

This diagram presents the SC4 standard development process in the future.

•An **ADM** (Application Data Model) is developed, potentially based on any number of **Functional Resources.**

•This methodology does neither require nor preclude the standardisation of ADMs.

•Such an ADM is very similar to the ARM in the current STEP Methodology, especially with respect to its level of abstraction. In contrast to the current STEP Methodology, there is a requirement, that an ADM must be sufficiently complete to support implementation, at least for validation purposes.

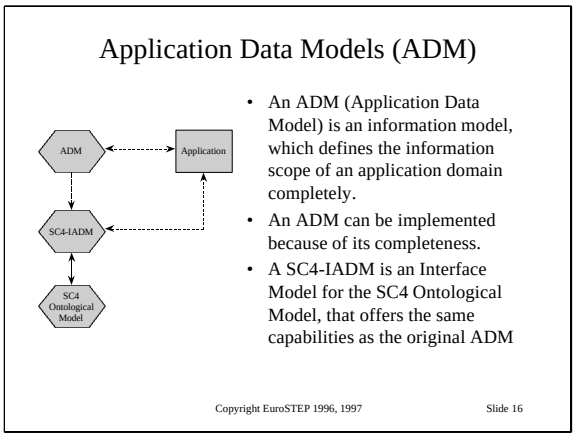
•Such an ADM may be **implemented** an put into application use.

•If there is a requirement to share information between the scope of this ADM and another one, the ADM will be converted into an **SC4-IADM** (Interoperable ADM) by adding the **model mapping** between the ADM and the **SC4 Ontological Model**.

•Under ideal circumstances, the SC4-IADM will be equivalent to the original ADM, and can be put into application use immediately.

•More realistically, there will be areas in the ADM that cannot be mapped to the SC4 Ontological Model. We may decide to leave it this way, accepting that the SC4-IADM will not support all applications that the original ADM did.

•Alternatively, we may create **SC4-OM Extensions** (SC4 Ontological Model Extensions) that we will integrate into the SC4 Ontological Model in order to enable the required mapping. This should have no impact on then existing parts of SC4 standards, as long as the SC4-OM Extensions belong to the context of the SC4 Ontological Model.



ADM

Compared with the current STEP Architecture/Methodology, an ADM here is a special case. Here we mean what is currently discussed as **Implementable ARM** in the STEP community.

From its intent, a better term for ADM would be “Application Information Model”. It has not been chosen, because its abbreviation “AIM” would be ambiguous, as “AIM” in the current STEP Architecture stands for “Aplication Interpreted Model”.

SC4-IADM

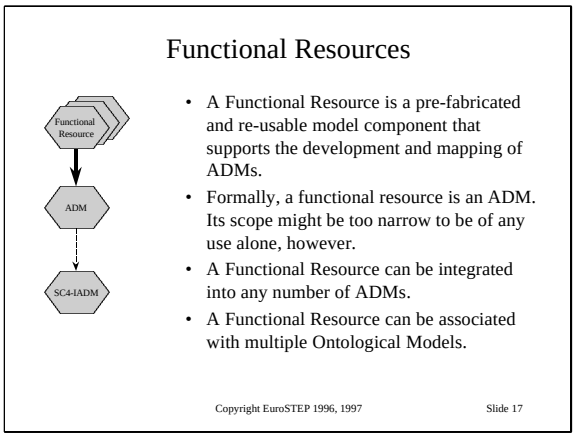
Compared with the current STEP Architecture/Methodology, a SC4-IADM is something totally different from the AIM, as it still preserves the original level of abstraction of the ARM. The closest you can get towards the current AIM in this new architecture is the set of all constructs in the SC4Ontological Model that is required to map the ADM.

SC4-IADM and SC4 Ontological Model

The mapping between a SC4-IADM and the SC4 Ontological Model must be a bi-directional instance mapping.

ADM, SC4-IADM, and SC4 Ontological Model

With this separation between the definition of the information scope and its mapping to the SC4 Ontological Model, it will be possible in the future, that a single ADM can be mapped to and used in different data exchange standards.



Functional Resource

An ADM may be based on any number of functional resources, pre-fabricated and re-usable model components that support the development and mapping of ADMs.

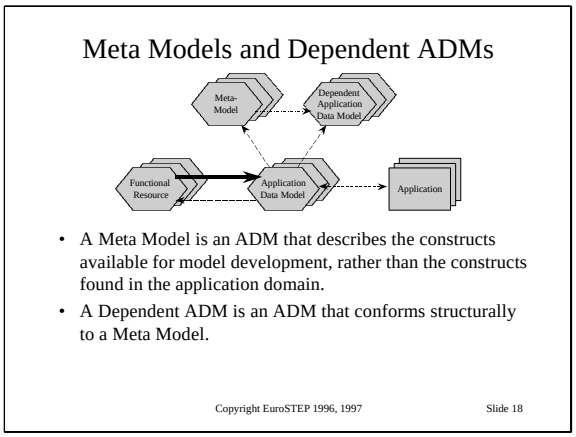
Formally, a functional resource is an ADM. However its scope may be too limited to be of any use alone.

Functional Resource, ADM, and SC4-IADM

In principle, every ADM and SC4-IADM can be used as or converted into a Functional Resource. However the maintainability and configuration control requirement for a complex family of standards, such as the SC4 standards, may add some constraints here.

Functional Resource in the current STEP Architecture/Methodology

The current STEP resources consist of some models that carry sufficient semantics to be used as they stand, such as part 42, and others that are more of the character of templates by intent. The first ones are likely candidates for the conversion into functional resources. The others may - potentially after some completions - become parts of the SC4 Ontological Model.



Meta Models

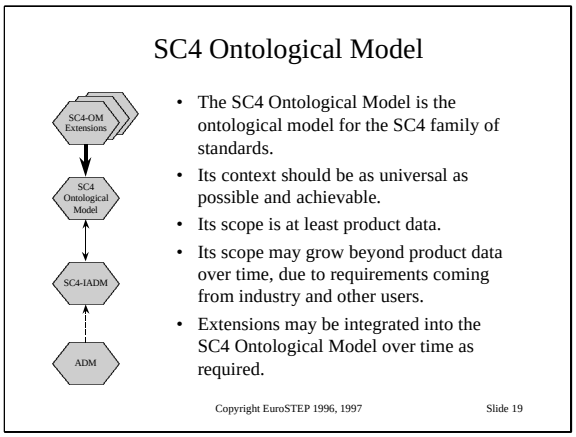
Sometimes, we cannot agree on the semantics of our models, but only on the structures from which we'd like to develop our models. Then we'll develop meta models. Meta models are ADMs in the sense, that we cannot use them, if there are no applications associated with them. However they are on higher levels of abstraction, especially with regard to structural abstraction.

Dependent ADMs

A Dependent ADM is an ADM that applies the constructs defined in a meta model.

Meta Model and DADM

As a dependent ADM can be a meta model at the same time, this structure is recursive and allows the definition of meta model hierarchies in an IRDS like manner.



SC4 Ontological Model in the current STEP Architecture/Methodology

There are 2 main sources for the development of the SC4 Ontological Model:

•The parts of the STEP Integrated Resources (40 series) that are on the right level of abstraction.

•The entity framework as proposed by Shell.

Current work in the area of computerised ontologies may contribute as well.

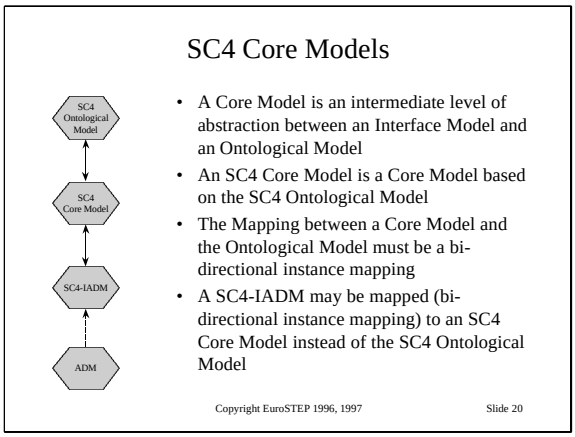
Ontological Model Evolution

The main reason, why this presentation prefers the term “model” over the term “schema”, is the intention to evolve the SC4 Ontological Model over time by adding new schemas without impacting the existing ones.

How do core models fit into this architecture?

In this architecture a core model is an SC4-IADM (more general an interface model) to which other SC4-IADMs are mapped as if it were an ontological model. Such a nested structure is not shown in the diagram, but possible in principle.

Some STEP Application Resources (100 series) may fall into this category as well.



SC4 Core Model

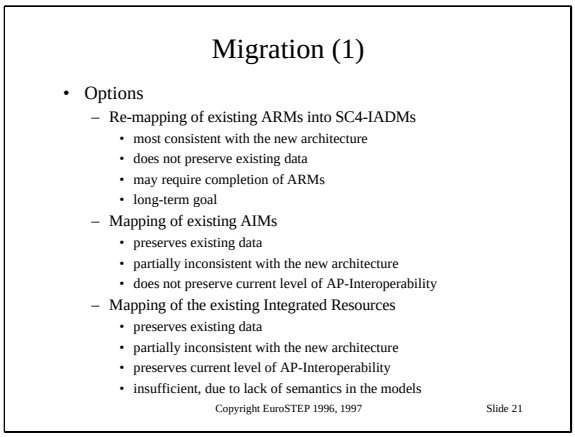
The concept of Core Models is not shown on the overview diagram.

Seen from the SC4 Ontological Model, an SC4 Core Model looks just like a SC4-IADM.

Seen from an SC4-IADM, a SC4 Core Model looks just like the SC4 Ontological Model.

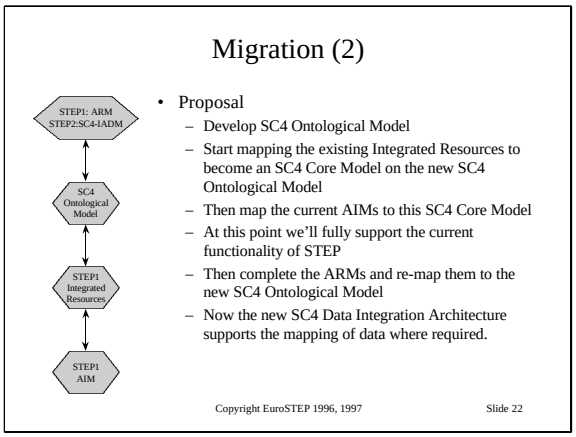
In principle, an SC4 Core Model is an SC4-IADM, which is used to integrate and harmonise other (dependent) SC4-IADMs.

An SC4 Core Model may depend on another SC4 Core Model instead of the SC4 Ontological Model. The SC4 Core Model for B&C (Building and Construction) may depend on the SC4 Core Model for AEC (Architecture, Engineering, and Construction) which in turn depends on the SC4 Ontological Model.



Mapping of existing AIMs

The statement, that this mapping might not preserve the existing level of AP interoperability, may be surprising. It is based on the assumption, that the existing ARM and mapping table will be consulted during the development of the mapping. In this case, an element existing in 2 AIMs may be mapped on different elements in the SC4 Ontological Model, because of their semantic differences.



Proposal:

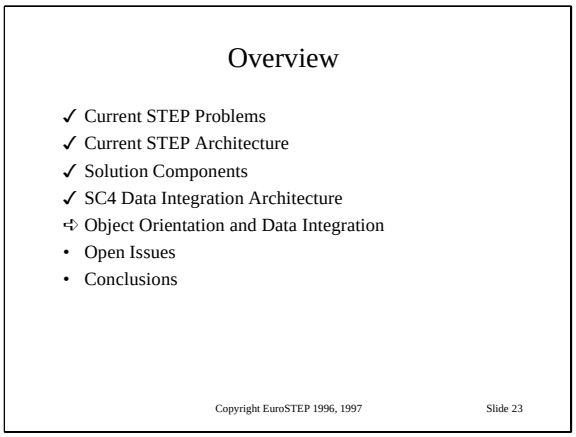
Some parts of the STEP Integrated resources, such as the management resources, will need to be completed before.

This proposal is based on the fact that data stored according to one SC4-IADM can be translated into data according to a different SC4-IADM through the SC4 Ontological Model.

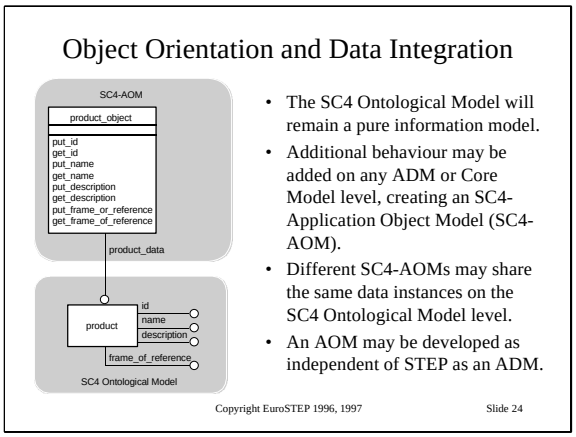
Here the first ADM is the existing STEP AIM, which is transformed into an SC4-IADM.

The 2nd SC4-IADM is the one derived from the existing ARM of our AP.

Of course, the completion and re-mapping of STEP1-ARMs will only be done, where required by potential users and where economically feasible.



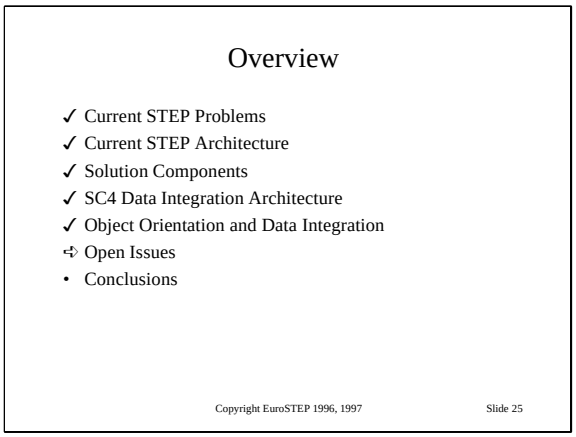
So far, we have covered the data aspects, we have not looked into problem of lack of behavioural modelling. This is what we are going for now.



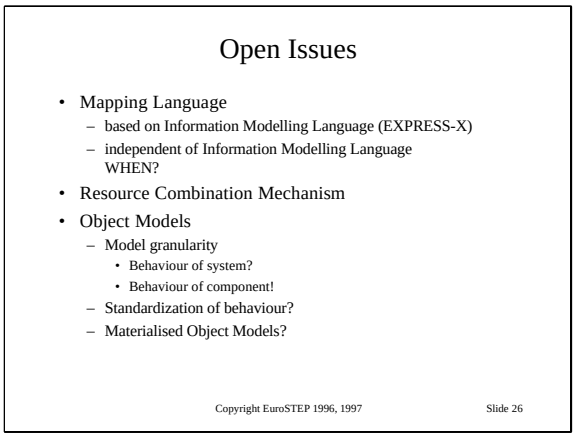
This diagram uses EXPRESS-G for the representation of the SC4 Ontological Model at the bottom and OMT for the representation of an SC4-AOM on top. To examplify the priciple structure, both models contain just a single entity or object respectively. For the same reason, the SC4-AOM shows only the most limited kind of behaviour.

An excerpt from the current STEP Part 41 is used here as an example of the future SC4 Ontological Model, despite the fact that its structure is not sufficiently semantically irreducible.

This proposal violates the **encapsulation principle** of object orientation in order to enable information sharing. This technique, which does not require the creation of a "supersystem" around 2 systems to be integrated, is sometimes called **virtual object orientation**, sometimes **proxy objects**.



What we have seen so far is a pretty consistent new architecture covering both data and behaviour aspects. Neverhteless, it would be unfair not to mention a couple of issues that are still open.



There are 3 categories of open issues:

Mapping Language

Nothing has been said here, that requires to use the same modelling technique for all these models. On the other side, an EXPRESS based mapping language (EXPRESS-X) will become available much earlier than a generic one. A sufficiently powerful generic mapping language will not become available before 2010, as its development is still requiring a significant amount of research.

Resource Combination Mechanism

This architecture could be enhanced significantly, if a simple, easy to use, declarative mechanism to combine functional resources could be found. The development of an “ADM Algebra” is still a research topic.

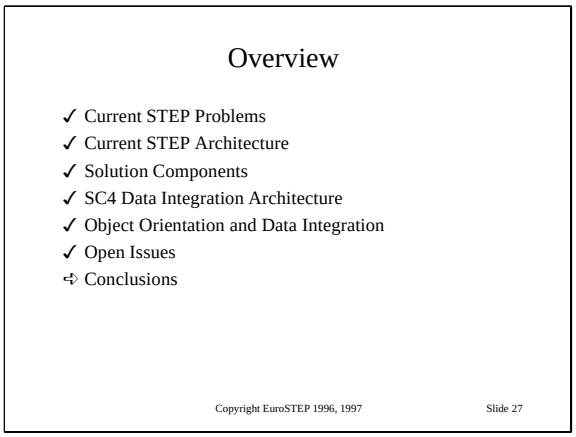
Object Models

The proposal presented in this presentation deals with the description of the behaviour of system components. It does not solve the problem of the description of the behaviour of whole systems. A solution for the latter is available from OMG (Object Management Group) with CORBA (Common Request Broker Architecture) and IDL (Interface Definition Language), on the other hand.

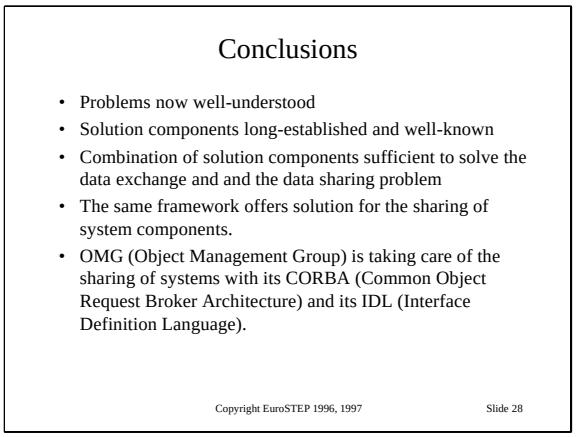
Another issue is the extent to which object behaviour should be standardised. No experience is available to guide such a decision to date.

The last open issue is whether SC4-AOMs must be materialised (persistent) or whether they can be generated on the fly (at run time). Such a decision will most probably be dependent on

- the structure of the SC4 Ontological Model,
- the structure of the SC4-AOM to be implemented, and
- the performance requirements of the applications to be supported.



After having covered the data aspects as well as the behavioural aspect of the proposed architecture, it's now time to draw some conclusions.



After more than 12 years of development, we now do understand the current STEP Architecture and Methodology, together with the problems it causes.

In addition, we can see, that such commonly known and well tested concepts as

•Conceptual/Ontological Modelling,

•ANSI-SPARC-Architecture, and ist derivate

•Data Exchange and Sharing Architecture,

•Object Orientation, and

•Virtual Object Orientation / Proxy Objects

can be fit together to create a solution to all known problems of the STEP architecture and methodology. The flexibility of the solution as sketched here also makes us confident, that the further development of the SC4 family of standards will require adaptations of this architecture, rather than a completely new one.